Package 'movedesign'

July 30, 2025

Title Study Design Toolbox for Movement Ecology Studies

Version 0.3.2

Maintainer Inês Silva <i.simoes-silva@hzdr.de>

Description Toolbox and 'shiny' application to help researchers design movement ecology studies, focusing on two key objectives: estimating home range areas, and estimating fine-scale movement behavior, specifically speed and distance traveled. It provides interactive simulations and methodological guidance to support study planning and decision-making. The application is described in Silva et al. (2023) <doi:10.1111/2041-210X.14153>.

License GPL (>= 3)

URL https://ecoisilva.github.io/movedesign/, https://ecoisilva.r-universe.dev/movedesign

BugReports https://github.com/ecoisilva/movedesign/issues

Depends R (>= 3.5.0)

Imports bayestestR, bsplus, combinat, config (>= 0.3.1), crayon, ctmm (>= 0.6.1), data.table, dplyr, fontawesome, gdtools, ggiraph, ggplot2, ggpubr, ggtext, golem (>= 0.3.2), grDevices, gsl, lubridate, parallel, parsedate, quarto, reactable, rintrojs, rlang, scales, shiny (>= 1.7.1), shinyalert, shinybusy, shinydashboard, shinydashboardPlus, shinyFeedback, shinyjs, shinyWidgets, stats, stringr, terra, tidyr, utils, viridis

Suggests knitr, rmarkdown, shinytest2

VignetteBuilder knitr, quarto

Config/testthat/edition 3

Encoding UTF-8

Language en-US

LazyData true

RoxygenNote 7.3.2

NeedsCompilation no

Author Inês Silva [cre, aut, cph] (ORCID:
<https://orcid.org/0000-0003-4850-6193>)

Repository CRAN

Date/Publication 2025-07-30 16:30:09 UTC

Contents

`fixrates` *Fix rates of animal tracking devices.*

Description

A dataset listing typical GPS fix rates for animal tracking devices. Useful for selecting typical sampling schedules in wildlife tracking projects.

Usage

```
fixrates
```

Format

A data.frame with 40 rows and 7 variables:

dti_notes Human-readable fix schedule, e.g., "1 fix every month". Helps interpret sampling intervals in practical terms.

dti Sampling interval in seconds, i.e., time between consecutive location fixes.

frq Sampling frequency in seconds, i.e., how often a fix occurs (inverse of dti).

frq_hrs Sampling frequency in hours, offering a more intuitive unit for comparison.

highlighted Logical. TRUE if the fix rate is commonly used in animal tracking studies. Useful for identifying standard settings. ...

`md_check` *Assess output convergence in simulation outputs*

Description

Evaluates whether the cumulative mean of a tracked error metric in simulation outputs has stabilized, indicating convergence. This function helps determine if repeated simulations or resampling have produced stable estimates, which is critical for reliable inference in animal movement projects.

Use this function after running `md_run()` or `md_replicate()` to check the reliability of outputs before further interpretation or reporting.

Usage

```
md_check(
  obj,
  m = NULL,
  tol = 0.05,
  n_converge = 10,
  plot = TRUE,
  pal = c("#007d80", "#A12C3B")
)
```

Arguments

`obj`	A `movedesign` or related object returned by `md_run()` or `md_replicate()`.
`m`	Numeric (optional). If provided, restricts the convergence check to results for a specific population sample size (`m`). Defaults to `NULL`, which checks up to the maximum population sample size.
`tol`	Numeric. The tolerance threshold for absolute change in the cumulative mean to declare convergence. Defaults to `0.05`.
`n_converge`	Integer. Number of consecutive steps within tolerance required to confirm convergence. Defaults to `10`.
`plot`	Logical. If `TRUE` (default), generates a plot of stepwise changes in the cumulative mean, highlighting when convergence is achieved.
`pal`	Character vector of color(s) of the plot, such as `c("#007d80", "#A12C3B")` (default).

Details

The cumulative mean of error is calculated, and the absolute changes over the last `n_converge` steps are inspected. If all are below the specified tolerance, convergence is declared.

If `plot = TRUE`, a plot is shown of absolute stepwise change in the cumulative mean, with a shaded region indicating the convergence threshold, aiding visual assessment.

Value

An object of class `"movedesign_check"` with the following elements:

`has_converged` Logical scalar indicating whether convergence was achieved.

`recent_deltas` Numeric vector of absolute changes in cumulative mean over the last `n_converge` steps.

`max_delta` Maximum absolute change among the last steps.

`tolerance` Numeric, the input tolerance `tol`.

`n_converge` Integer, the input `n_converge`.

variable Character. Name of the variable checked.

recent_cummean Numeric vector. The last cumulative means checked.

See Also

`md_run()`, `md_replicate()`

Examples

```
if(interactive()) {
  # After running a simulation or resampling:
  md_check(output, tol = 0.05, n_converge = 10)
}
```

`md_configure` *Interactively configure movement design setup*

Description

Guides the user to assign each argument required for a movement design workflow, including species label and key simulation settings. Users may choose to set a specific population sample size (number of animals tagged/to be tagged) or optimize the population sample size considering a specific analytical target.

Usage

```
md_configure(data, models = NULL)
```

Arguments

data	A named list of telemetry objects (from ctmm::as.telemetry()) to be used as the empirical basis for the simulations. Each telemetry object must contain valid metadata and timestamped locations.
models	(Optional) Named list of fitted ctmm models (from ctmm::ctmm.fit() or ctmm::ctmm.select()). If not supplied, models are fitted automatically.

Details

The argument data is **required** and must be supplied directly (as a list of telemetry objects, obtained from ctmm::as.telemetry()). The argument models is optional, and if omitted, models will be fitted automatically.

Value

An object of class movedesign_input (and movedesign). This is a structured S3 list containing all validated inputs, model fits, and derived parameters for the study design workflow.

See Also

md_prepare()

Examples

```
if(interactive()) {
  data(buffalo)
  md_params <- md_configure(data = buffalo)
}
```

---

md_merge *Merge multiple simulation outputs*

Description

Merges the results of multiple simulation runs, each produced by md_run(), into a single unified movedesign_output object. This is especially useful when running replicate simulations for power analyses, sensitivity testing, or batch processing. Merging allows you to aggregate all simulated individuals, outputs, and related metadata, enabling streamlined downstream analyses.

Usage

```
md_merge(...)
```

Arguments

...	One or more objects of class `movedesign_preprocess`, typically generated by `md_run()`. Each object must contain, at minimum, the elements `simList`, `simfitList`, and `seedList`. Optional elements such as `akdeList` and `ctsdList` are merged if present.

Value

A single `movedesign_output` object that contains all merged simulation outputs and inherits metadata from the first input object. The output includes:

- Merged list of simulated individuals (`simList`),
- Merged list of fitted models (`simfitList`),
- Merged list of seeds used for each simulation replicate (`seedList`),
- Optionally, merged home range (`akdeList`) and speed (`ctsdList`) outputs,
- Relevant metadata describing the study design parameters.

See Also

`md_prepare`, `md_run`

Examples

```
if (interactive()) {
  input <- md_prepare(
    data = buffalo,
    models = models,
    species = "buffalo",
    n_individuals = 5,
    dur = list(value = 1, unit = "month"),
    dti = list(value = 1, unit = "day"),
    add_individual_variation = TRUE,
    grouped = FALSE,
    set_target = "hr",
    which_meta = "mean"
  )

  output1 <- md_run(input)
  output2 <- md_run(input)

  merged <- md_merge(output1, output2)
}
```

`md_optimize` *Optimize population sample size and sampling parameters*

Description

Repeatedly simulates movement datasets across a range of candidate population sample sizes to identify the minimal sample size and associated sampling parameters (e.g., duration, sampling interval) required to achieve a predefined error threshold for key space-use and movement metrics (home range area, or speed).

The function quantifies estimation error for each metric and sample size, evaluates which population sample size reliably meets target thresholds, and reports final recommendations.

Usage

```
md_optimize(
  obj,
  n_replicates = NULL,
  error_threshold = NULL,
  verbose = FALSE,
  trace = TRUE,
  parallel = FALSE,
  ncores = parallel::detectCores(),
  plot = FALSE,
  ...
)
```

Arguments

`obj`	A movement design input object (see `md_prepare()` or `md_configure()`).
`n_replicates`	Integer. Number of simulation replicates at each candidate sample size.
`error_threshold`	Numeric. Error threshold (e.g. `0.05` for 5%) to display as a reference in the plot.
`verbose`	Logical. If `TRUE` (default), prints a summary of the convergence check to the console.
`trace`	Logical; if `TRUE` (default), prints progress and timing messages to the console.
`parallel`	Logical; if `TRUE`, enables parallel processing. Default is `FALSE`.
`ncores`	Integer; number of CPU cores to use for parallel processing. Defaults to all available cores detected by `parallel::detectCores()`.
`plot`	Logical. If TRUE, displays a diagnostic plot of the final results.
`...`	Additional arguments used internally.

Details

The function iteratively runs movement design simulations for increasing population sample sizes (`m`), evaluating error for each replicate and metric via meta-analyses. Convergence is checked using the error threshold and stability of cumulative mean error. The function stops when a sample size meets all criteria or the maximum population sample size is reached. Results can be visualized using if `plot = TRUE`.

Value

A list of class `movedesign_report` containing:

- `summary`: Data frame of summary statistics for each replicate, sample size, and metric.
- `error_threshold`: Numeric. The error threshold used.
- `sampling_duration`: Character string. Final sampling duration.
- `sampling_interval`: Character string. Final sampling interval.
- `sample_size_achieved`: Logical. Indicates if convergence was achieved and the threshold met.
- `minimum_population_sample_size`: Integer. Minimum sample size achieving the threshold (or maximum evaluated if `sample_size_achieved` is `FALSE`).

Note

Some biologgers inherently involve a trade-off between fix frequency and battery life. Shorter intervals between location fixes offer higher temporal resolution but reduce deployment duration due to increased power consumption. In contrast, longer deployments require less frequent sampling to conserve battery.

This trade-off makes it challenging to estimate multiple metrics with differing data needs: high-resolution data (shorter intervals) improve speed estimation, while extended deployments (longer durations) are vital for accurate home range area estimates. A sampling design that minimizes error for one metric may increase error for another.

Researchers facing these constraints should consider prioritizing a single research target (e.g., either home range area *or* speed), or use stratified designs to balance data needs across individuals.

See Also

`md_prepare()`, `md_configure()`

Examples

```
if(interactive()) {
  obj <- md_configure(data = buffalo,
                      models = models)

  out <- md_optimize(tmp,
                     n_replicates = 25,
                     error_threshold = 0.05,
                     plot = TRUE)
}
```

`md_plot` *Visualize study design outputs*

Description

Produces a publication-ready density plot showing the distribution of relative error estimates from study design simulations. The plot highlights the mean and a shaded credible interval (CI) region, following the computation of credible intervals as implemented in `bayestestR::ci()`. If groups are present, density curves for each group are overlaid for comparison, using customizable colors.

This function is typically used after running `md_replicate()`, providing a visual diagnostic of simulation results.

Usage

```
md_plot(
  obj,
  ci = 0.95,
  method = "HDI",
  pal = c("#007d80", "#A12C3B"),
  m = NULL
)
```

Arguments

`obj`	A `movedesign_output` object, as returned by `md_replicate()`. The object must contain a `summary` data frame with, at a minimum, the following columns: **error** Relative error values for each replicate. **error_lci** Lower credible interval bound for error. **error_uci** Upper credible interval bound for error. **group** (Optional) Group label for comparing densities.
`ci`	Numeric scalar between 0 and 1. The probability of the credible interval (CI) to be estimated. Default to `0.95` (95%).
`method`	Character. Credible interval estimation method (passed to `bayestestR::ci()`; default: `"HDI"`). See `?bayestestR::ci()` for more details.
`pal`	Character vector of color(s) for the density, CI shading, and mean line. If a single group, supply one color (default: `"#007d80"`). If groups are present, supply two colors (default: `c("#007d80", "#A12C3B")`).
`m`	Numeric (optional). If provided, restricts the results for a specific population sample size (`m`). Defaults to `NULL`, which checks up to the maximum population sample size.

Details

This plot helps users assess the reliability of simulation outputs by visualizing the distribution of relative errors. When multiple groups are simulated, the plot enables direct visual comparison of performance across groups. If credible intervals cannot be calculated, a warning is issued and only the density curves are displayed.

It is strongly recommended to use `md_check()` to assess whether the distributions shown here have stabilized. Checking for convergence ensures that the summary statistics and uncertainty estimates depicted in the plot are reliable and not unduly influenced by too few replicates or ongoing variability. Running `md_check()` helps you determine if additional simulation replicates are needed to achieve stable inference in your design evaluation.

Value

A `ggplot` object showing:

- Density curve(s) of the relative error distribution.
- Shaded region for the central credible interval.
- Vertical dashed lines at mean(s).
- Overlaid densities if multiple groups are present.
- Percent-formatted x-axis for interpretation.

This object can be further customized with additional `ggplot2` layers if needed.

See Also

`md_replicate()`, `md_check()` for convergence diagnostics, and refer to `bayestestR::ci()` for details on credible interval computation and interpretation.

Examples

```
if (interactive()) {
  input <- md_prepare(
    data = buffalo,
    models = models,
    species = "buffalo",
    n_individuals = 5,
    dur = list(value = 1, unit = "month"),
    dti = list(value = 1, unit = "day"),
    add_individual_variation = TRUE,
    grouped = TRUE,
    set_target = "hr",
    which_meta = "mean"
  )

  output <- md_replicate(input, n_replicates = 20)

  # Plot with 80% credible intervals:
  md_plot(output, ci = 0.80, method = "HDI")
}
```

`md_plot_preview` *Preview plot for movedesign workflow outputs (single replicate)*

Description

Generates a quick visualization of relative error for home range or movement speed estimation from a single replicate of a movedesign workflow. This functions shows preliminary outputs only and should not be used to evaluate study design.

Use the output of `md_run()` (a `movedesign_preprocess` object). Users should run `md_replicate()` for a full assessment.

Usage

```
md_plot_preview(obj, error_threshold = 0.05, pal = c("#007d80", "#A12C3B"))
```

Arguments

`obj`	An object of class `movedesign_preprocess` (output of `md_run()`).
`error_threshold`	Numeric. Error threshold (e.g. `0.05` for 5%) to display as a reference in the plot.
`pal`	Character vector of two colors for within/outside threshold (default: c("#007d80", "#A12C3B")).

Details

For robust results and credible intervals, use `md_replicate()`.

Value

A ggplot object displaying relative error by population sample size, with point estimate and confidence intervals for mean estimates, and horizontal error threshold lines.

See Also

`md_run()`, `md_replicate()`

Examples

```
if (interactive()) {
  input <- md_prepare(
    data = buffalo,
    models = models,
    species = "buffalo",
    n_individuals = 5,
    dur = list(value = 1, unit = "month"),
    dti = list(value = 1, unit = "day"),
    add_individual_variation = TRUE,
```

```
    grouped = TRUE,
    set_target = "hr",
    which_meta = "mean"
  )

  output <- md_run(input)
  md_plot_preview(output, error_threshold = 0.05)
}
```

`md_prepare` *Prepare movement study design inputs*

Description

Prepares, validates, and organizes all required inputs and parameters for evaluating the study design of animal movement projects. This function checks data inputs, fits or verifies movement models, extracts key parameters, and consolidates all settings in a structured object for easy and reproducible downstream use.

Usage

```
md_prepare(
  species = NULL,
  data,
  models = NULL,
  n_individuals = NULL,
  dur = NULL,
  dti = NULL,
  set_target = c("hr", "ctsd"),
  which_meta = "mean",
  add_individual_variation = FALSE,
  groups = NULL,
  parallel = FALSE
)
```

Arguments

`species`	Character. Scientific or common name of the focal species used as a workflow label.
`data`	A named list of telemetry objects (from `ctmm::as.telemetry()`) to be used as the empirical basis for the simulations. Each telemetry object must contain valid metadata and timestamped locations.
`models`	(Optional) Named list of fitted ctmm models (from `ctmm::ctmm.fit()` or `ctmm::ctmm.select()`). If not supplied, models are fitted automatically.
`n_individuals`	Integer. Number of animals (tags) to include in the study design; defines the target *population* sample size.

dur — A list with elements value and unit (e.g., list(value = 2, unit = "months")), for the study's maximum duration. unit must be either "second", "minute", "hour", "day", "month", or "year".

dti — A list with elements value and unit (e.g., list(value = 1, unit = "day")), specifying the intended sampling interval between relocations. unit must be either "second", "minute", "hour", "day", "month", or "year".

set_target — Character. Specifies the primary research target(s): must be either hr (home range estimation), ctsd (movement speed), or a character vector including both. This argument controls which target metrics are processed, analyzed, and reported in the study design workflow.

which_meta — Character. Specifies the analytical target for population-level inference: NULL, "mean" (default), or "ratio". Use NULL for a single individual, "mean" for population means, or "ratio" to compare group means (requires groups).

add_individual_variation — Logical. If TRUE, simulates variation by drawing movement parameters from the population distribution.

groups — (Optional) A named list for group assignments. Each element is a character vector of individual names (matching data). For example, list(A = c("id1", "id2"), B = c("id3", "id4")) for groups "A" and "B".Required when which_meta = \"ratio\".

parallel — Logical. If TRUE, enables parallel processing for model fitting, which speeds up analyses.

Details

This function is designed to streamline and standardize the preparation of input data and study design parameters for simulation-based movement ecology analyses. It performs the following key steps:

- Validates that data is a non-empty list of telemetry objects with metadata and location records.
- Fits movement models to each individual if not supplied.
- Checks supplied movement models for validity.
- Extracts parameters (e.g., sigma, tau_p, tau_v) for simulation.
- Gathers settings (sample size, duration, sampling, grouping) into a single object.

Value

An object of class movedesign_input (and movedesign). This is a structured S3 list containing all validated inputs, model fits, and derived parameters for the study design workflow.

The returned object includes:

- design: A movedesign object with all study settings and metadata.
- data: The original or validated list of telemetry objects.
- fitList: List of fitted movement models for each individual.
- meanfitList: List of population or group-level mean models.
- sigma, tau_p, tau_v: Movement parameters extracted from data provided for downstream simulations.

- mu: List of mean locations.
- groups: Group structure if specified, otherwise NULL.
- Other slots describing *population* sample size, sampling duration, sampling interval, targets, and workflow options.

This object is ready for use in downstream movedesign output and diagnostic functions.

Examples

```
if(interactive()) {
  data(buffalo)
  input <- md_prepare(
    data = buffalo,
    models = models,
    species = "buffalo",
    n_individuals = 5,
    dur = list(value = 1, unit = "month"),
    dti = list(value = 1, unit = "day"),
    add_individual_variation = TRUE,
    set_target = "hr",
    which_meta = "mean")
 summary(input)
}
```

---

md_replicate *Replicate study design and aggregate simulation outputs*

Description

Runs the specified movement study design multiple times and aggregates outputs and summary statistics across independent replicates. This enables sensitivity analyses and quantifies variability arising from random sampling, especially when individual-level variation is enabled (i.e., add_individual_variation = TRUE in md_prepare()). Replication helps assess how stochasticity and design choices impact simulation inference.

Usage

```
md_replicate(
  obj,
  n_replicates,
  verbose = FALSE,
  trace = TRUE,
  parallel = FALSE,
  ncores = parallel::detectCores()
)
```

Arguments

obj	An object of class movedesign created by md_prepare(). It contains all parameters and input data defining the movement study.
n_replicates	Integer specifying how many independent simulation replicates to run.
verbose	Logical; if TRUE, runs population-level inferences iteratively for increasing population sample sizes, saving results at each step. Defaults to FALSE, which runs only once for the maximum sample size defined by n_individuals in md_prepare().
trace	Logical; if TRUE (default), prints progress and timing messages to the console.
parallel	Logical; if TRUE, enables parallel processing. Default is FALSE.
ncores	Integer; number of CPU cores to use for parallel processing. Defaults to all available cores detected by parallel::detectCores().

Details

Each replicate runs independently using the same study design object but with a unique random seed to ensure independence. Results from all replicates are merged using md_merge(), and summary statistics combine into a single data.table for convenient downstream analyses and evaluation. Parallel processing can significantly reduce runtime when running many replicates; use ncores to specify the number of CPU cores used. If function is interrupted (e.g., Ctrl+C), it returns results from all completed replicates up to that point.

Value

A list of class movedesign_output with two elements:

- data: A list containing merged simulation outputs from all replicates.
- summary: A data.table summarizing key statistics for each replicate.

See Also

md_prepare(), md_run(), md_merge()

Examples

```
if (interactive()) {
  input <- md_prepare(
    data = buffalo,
    models = models,
    species = "buffalo",
    n_individuals = 5,
    dur = list(value = 1, unit = "month"),
    dti = list(value = 1, unit = "day"),
    add_individual_variation = TRUE,
    grouped = FALSE,
    set_target = "hr",
    which_meta = "mean"
  )
```

```
  output <- md_replicate(input, n_replicates = 5)
}
```

`md_run` *Run study design workflow*

Description

Executes a complete simulation and analysis workflow for an animal movement study design prepared using `md_prepare()`. This function simulates telemetry data, fits movement models, estimates home ranges and/or movement speeds, and stores all results in the returned object. Progress and timing messages are printed by default.

Usage

```
md_run(design, trace = TRUE)
```

Arguments

`design`	An object of class `movedesign` (and `movedesign_input`), as returned by `md_prepare()`, containing all study design parameters and data.
`trace`	Logical. If TRUE (default), print progress and timing messages to the console.

Details

This function ensures reproducibility by saving all random seeds and intermediate results. Progress and timing messages help track the simulation workflow.

Typical workflow:

- Prepare a study design with `md_prepare()`.
- Run all simulations and analyses with `md_run()`.
- Summarize or plot outputs from the returned object.

Value

An updated `movedesign` object (subclass `movedesign_preprocess`) containing all simulation and outputs components:

- `simList`: List of simulated telemetry datasets, one per individual.
- `seedList`: List of random seeds used for reproducibility.
- `simfitList`: List of fitted movement models for each simulation.
- `akdeList`: List of home range (AKDE) estimates, present if the `hr` target was listed in `set_target`.
- `ctsdList`: List of continuous-time speed and distance (CTSD) estimates, present if the `ctsd` target was listed in `set_target`.

See Also

md_prepare(), md_replicate(), md_check(), md_plot()

Examples

```
if(interactive()) {
input <- md_prepare(
  data = buffalo,
  models = models,
  species = "buffalo",
  n_individuals = 5,
  dur = list(value = 1, unit = "month"),
  dti = list(value = 1, unit = "day"),
  add_individual_variation = TRUE,
  set_target = "hr",
  which_meta = "mean"
)
output <- md_run(input)
}
```

---

movmods *Table of movement processes.*

Description

Lists all continuous-time movement process models in **ctmm**. Each row is a different movement model applicable for animal movement.

Usage

```
movmods
```

Format

A `data.frame` with 5 rows and 6 variables:

name Full descriptive name of the model (e.g., "Ind. Ident. Distr. (IID)"). Used throughout `ctmm`. See reference for more details on each model and their properties.

name_short Abbreviated name, used where space is limited.

tau_p Logical. TRUE if the model includes the position autocorrelation timescale (i.e., home range crossing time).

tau_v Logical. TRUE if the model includes the velocity autocorrelation timescale (i.e., directional persistence).

hrange Logical; TRUE if the model supports range residency, meaning the animal is likely to remain within a bounded area or "home range" instead of expanding indefinitely.

pars Character string summarizing which autocorrelation parameters (e.g., tau_p, tau_v) the model estimates. Shown in HTML for documentation. ...

References

- Calabrese et al. (2016). ctmm: an R package for analyzing animal relocation data as a continuous-time stochastic process. Methods in Ecology and Evolution, 7(9), 1124-1132 doi:10.1111/2041-210X.12559.
- Silva et al. (2022). Autocorrelation-informed home range estimation: A review and practical guide. Methods in Ecology and Evolution, 13(3), 534-544 <10.1111/2041-210X.13786>.

`print.movedesign_check`

Print method for `movedesign_check` *objects*

Description

Print method for `movedesign_check` objects

Usage

```
## S3 method for class 'movedesign_check'
print(x, ...)
```

Arguments

`x`	An object of class `movedesign_check`
`...`	Unused

`print.movedesign_input`

Print method for `movedesign_input`

Description

Print method for `movedesign_input`

Usage

```
## S3 method for class 'movedesign_input'
print(x, ...)
```

Arguments

`x`	An object of class `movedesign_input`
`...`	Additional arguments

print.movedesign_output

Print method for movedesign_output

Description

Print method for movedesign_output

Usage

```
## S3 method for class 'movedesign_output'
print(x, ...)
```

Arguments

x	An object of class movedesign_output
...	Additional arguments

print.movedesign_preprocess

Print method for movedesign_preprocess

Description

Print method for movedesign_preprocess

Usage

```
## S3 method for class 'movedesign_preprocess'
print(x, ...)
```

Arguments

x	An object of class movedesign_preprocess
...	Additional arguments

print.movedesign_report *Print method for* movedesign_report *objects*

Description

Print method for movedesign_report objects

Usage

```
## S3 method for class 'movedesign_report'
print(x, ...)
```

Arguments

x	An object of class movedesign_report
...	Unused

run_app *Run movedesign R Shiny application*

Description

Run movedesign R Shiny application

Usage

```
run_app(
  onStart = NULL,
  options = list(),
  enableBookmarking = NULL,
  uiPattern = "/",
  ...
)
```

Arguments

onStart	A function called before the app runs. Only relevant for programmatic usage.
options	A named list passed to shiny::runApp.
enableBookmarking	One of url, server, or disable. The default, NULL, respects any previous call to enableBookmarking.
uiPattern	A regular expression used to match request paths. The request path must match the expression in full to be handled by the UI.
...	arguments to pass to golem_opts. See ?golem::get_golem_options for more details.

Value

No return value. This function is called for its side effects.

`summary.movedesign_check`
Summary method for `movedesign_check` *objects*

Description

Summary method for `movedesign_check` objects

Usage

```
## S3 method for class 'movedesign_check'
summary(object, ...)
```

Arguments

`object`	An object of class `movedesign_check`
`...`	Unused

`summary.movedesign_input`
Summary method for `movedesign_input`

Description

Summary method for `movedesign_input`

Usage

```
## S3 method for class 'movedesign_input'
summary(object, ...)
```

Arguments

`object`	An object of class `movedesign_input`
`...`	Additional arguments

summary.movedesign_output *Summary method for* movedesign_output

Description

Summary method for movedesign_output

Usage

```
## S3 method for class 'movedesign_output'
summary(object, ...)
```

Arguments

object	An object of class movedesign_output
...	Additional arguments

summary.movedesign_preprocess *Summary method for* movedesign_preprocess

Description

Summary method for movedesign_preprocess

Usage

```
## S3 method for class 'movedesign_preprocess'
summary(object, ...)
```

Arguments

object	An object of class movedesign_preprocess
...	Additional arguments

summary.movedesign_report *Summary method for* movedesign_report *objects*

Description

Summary method for movedesign_report objects

Usage

```
## S3 method for class 'movedesign_report'
summary(object, ...)
```

Arguments

object	An object of class movedesign_report
...	Unused

Index